

Title	Grammar-based compression approach to extraction of common rules among multiple trees of glycans and RNAs.
Author(s)	Zhao, Yang; Hayashida, Morihiro; Cao, Yue; Hwang, Jaewook; Akutsu, Tatsuya
Citation	BMC bioinformatics (2015), 16
Issue Date	2015-04-24
URL	<a href="http://hdl.handle.net/2433/210405">http://hdl.handle.net/2433/210405</a>
Right	© Zhao et al.; licensee BioMed Central. 2015; This is an Open Access article distributed under the terms of the Creative Commons Attribution License ( <a href="http://creativecommons.org/licenses/by/4.0">http://creativecommons.org/licenses/by/4.0</a> ), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The Creative Commons Public Domain Dedication waiver ( <a href="http://creativecommons.org/publicdomain/zero/1.0/">http://creativecommons.org/publicdomain/zero/1.0/</a> ) applies to the data made available in this article, unless otherwise stated.
Type	Journal Article
Textversion	publisher

RESEARCH ARTICLE

Open Access

# Grammar-based compression approach to extraction of common rules among multiple trees of glycans and RNAs

Yang Zhao, Morihito Hayashida<sup>\*</sup>, Yue Cao, Jaewook Hwang and Tatsuya Akutsu

## Abstract

**Background:** Many tree structures are found in nature and organisms. Such trees are believed to be constructed on the basis of certain rules. We have previously developed grammar-based compression methods for ordered and unordered single trees, based on bisection-type tree grammars. Here, these methods find construction rules for one single tree. On the other hand, specified construction rules can be utilized to generate multiple similar trees.

**Results:** Therefore, in this paper, we develop novel methods to discover common rules for the construction of multiple distinct trees, by improving and extending the previous methods using integer programming. We apply our proposed methods to several sets of glycans and RNA secondary structures, which play important roles in cellular systems, and can be regarded as tree structures. The results suggest that our method can be successfully applied to determining the minimum grammar and several common rules among glycans and RNAs.

**Conclusions:** We propose integer programming-based methods MinSEOTGMul and MinSEUTGMul for the determination of the minimum grammars constructing multiple ordered and unordered trees, respectively. The proposed methods can provide clues for the determination of hierarchical structures contained in tree-structured biological data, beyond the extraction of frequent patterns.

**Keywords:** Grammar-based compression, Bisection-type tree grammar, Glycan, RNA secondary structure

## Background

Many tree structures are found in nature and organisms. One such tree structure is a glycan, in which each monosaccharide is regarded as a vertex, except for cyclic oligosaccharides and so on. Since glycans contain various complicated structures, they are believed to be constructed by various mechanisms that recognize a monosaccharide, binding it with another. For instance, a galactosyltransferase is known to catalyze the biosynthesis with a galactose [1]. Glycans are also known to play several important roles in a cellular system, such as molecular recognition, cell adhesion, and antigen-antibody interactions. Therefore, many studies have been conducted to understand the structures and functions of glycans; in addition, several methods have been developed for the

discovery of glycan motifs or significant subtrees, as glycan structures are conserved in evolutionary processes [2-4], and to measure the similarities between glycans [5,6].

RNA secondary structures can be also regarded as tree structures; these structures depend on the nucleic acid sequence. RNAs, which are large biological molecules, also perform important functions in living cells, such as the catalysis of biological reactions and expression of genes. Milo et al. displayed a pseudoknot-free RNA secondary structure as an ordered rooted tree, wherein each base pair, unpaired base interval, hairpin loop, internal loop, multi-loop, and external loop corresponds to a vertex, and developed a cubic time algorithm for the homeomorphic subtree alignment problem [7]. They applied it to pairwise alignments of RNA secondary structures, and found several structural similarities, which were not detected by other existing algorithms. Chen and Zhang developed an efficient algorithm for tree edit distance, and

<sup>\*</sup>Correspondence: morihito@kuicr.kyoto-u.ac.jp  
Bioinformatics Center, Institute for Chemical Research, Kyoto University,  
Gokasho, Uji, Japan

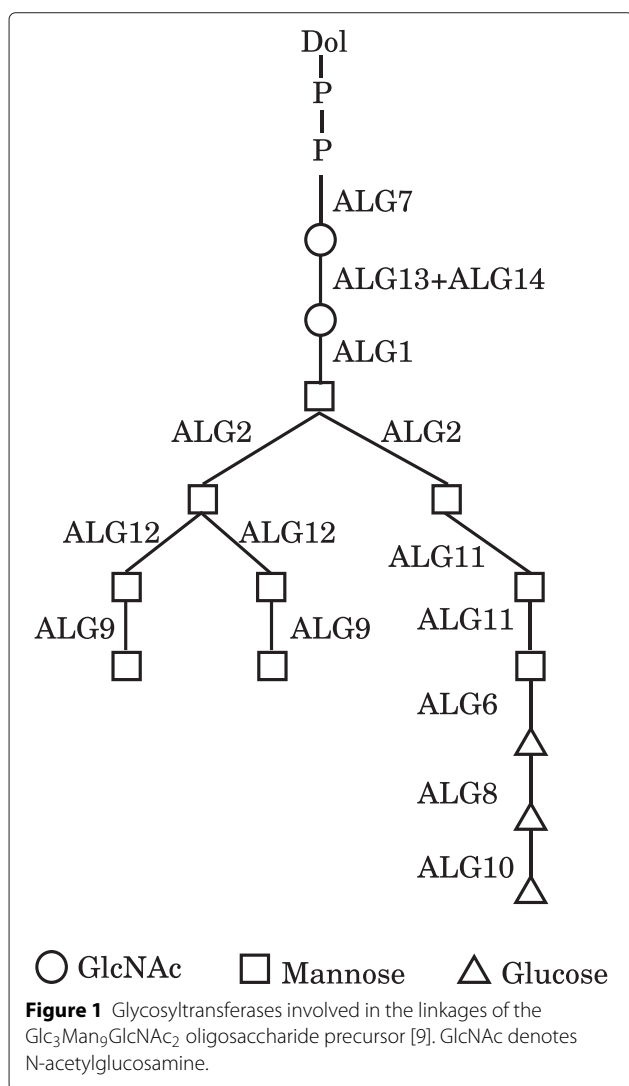
used the same to compare several RNA secondary structures [8]. These methods tried to measure the similarities between the tree structures, and to determine frequent subtrees.

In this paper, we focus on finding construction rules for multiple biomolecular tree structures. For example, it was reported that the glycosyltransferases such as ALG1 and ALG2 are involved in the linkages of  $\text{Glc}_3\text{Man}_9\text{GlcNAc}_2$  oligosaccharide precursor as shown in Figure 1 [9], where Glc, Man, and GlcNAc stand for glucose, mannose, and N-acetylglucosamine, respectively. ALG1 connects a GlcNAc with a Man, and ALG2 connects a Man with the Man connected by ALG1. According to local structures, different glycosyltransferases catalyze those biosyntheses in order to construct the same structure of the oligosaccharide. Since it is difficult to find existences of such genes from one tree structure, we try to find them from

multiple tree structures that the same enzyme constructs a specified local structure.

In the field of computer science, grammar-based compression is used to determine the rules of construction of various types of data. The identification of the smallest grammar for input data would provide a clue towards understanding the construction rules. The determination of the smallest Context-Free Grammar (CFG) constructing a given string is known to be NP-hard [10]. Polynomial-time approximation methods have been developed to determine the smallest CFG for sequence data such as DNA, RNA nucleic acid sequences, and protein amino acid sequences [10-13]. Although these methods were extended to the tree structured data including XML [14-16], the minimum grammar has not always been provided. Hence, in our previous study, we developed grammar-based compression methods for single trees that always output the minimum grammar. We used bisection-type tree grammars proposed in [17], Simple Elementary Ordered Tree Grammar (SEOTG) for ordered trees and Simple Elementary Unordered Tree Grammar (SEUTG) for unordered trees, in which at most two nonterminal and terminal symbols appear on the right-hand side of each production rule. Since it seemed to be difficult to directly formulate the problems of finding the minimum SEOTG and SEUTG using Integer Programming (IP), we instead formulated the problems of finding SEOTG and SEUTG with a given size [18].

We previously developed methods for compressing single trees. As considered in the example provided in Figure 1, specified construction rules can be applied to generate several similar trees. Therefore, in this study, we attempt to discover common construction rules among multiple distinct trees; in addition, by improving and extending the previous methods, we propose the novel IP formulations MinSEOTGMul and MinSEUTGMul, for the direct determination of the minimum grammars of SEOTG and SEUTG. In our previous study, the problems associated with the determination of the minimum SEOTG and SEUTG were not directly formulated by IP; instead, the problems associated with determining the SEOTG and SEUTG utilizing the given sizes were formulated. Therefore, the previous IP was executed at least twice with different parameters in order to confirm the minimum size of the grammars that construct a given single tree. The methods proposed in this paper can be applied to the direct determination of the minimum grammar in one attempt. As for multiple input trees, our previous method can be trivially extended for the determination of the minimum SEOTG and SEUTG for  $N$  trees, which adds a special root vertex, connects each root of  $N$  trees to the special root by a distinct special edge, and applies the previous IP formulation to this generated single tree. This approach, however, increases the number



of variables in the IP formulation, and may enlarge the execution time.

We apply our methods to several sets of glycans and RNA secondary structures. Consequently, we successfully determined the minimum grammar and several common construction rules using this method.

**Methods**

In this section, we briefly review the CFG, SEOTG, and SEUTG [17], and explain the proposed IP formulations, MinSEOTGMul and MinSEUTGMul, for multiple trees. Integer Programming (IP) is a method used to optimize a linear objective function subject to linear inequality constraints, with the variables being restricted to integers. We use these tools to solve the proposed integer program as our problem of finding the minimum SEOTG and SEUTG is NP-hard, with efficient solvers being developed. The benefit to use IP is that we can obtain exact solutions for combinatorial optimization problems.

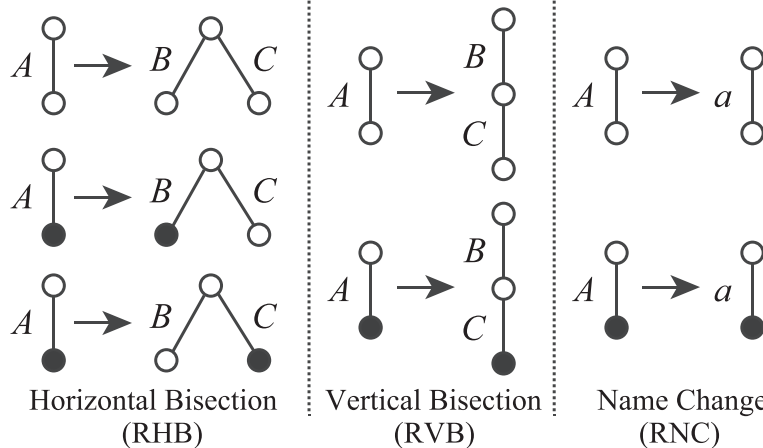
**Context-free grammar (CFG)**

A context-free grammar (CFG) deals with strings, and is represented by 4-tuple  $(T, V, S, P)$ , where  $T$  is a set of terminal symbols,  $V$  is a set of nonterminal symbols,  $S$  is a start symbol in  $V$ , and  $P$  is a set of production rules, wherein a nonterminal symbol on the left-hand side is replaced with a string on the right-hand side, which consists of symbols from  $V$  and  $T$  [19]. The final product generated by a CFG does not include any nonterminal symbol. The size of a grammar is defined as the total number of symbols appearing on the right-hand side of the production rules. For example, in a case where  $T = \{a, b\}$ ,  $V = \{S\}$ , and  $P = \{S \rightarrow aSb, S \rightarrow ab\}$ , the start symbol  $S$  is repeatedly replaced with the rule  $S \rightarrow aSb$ , all

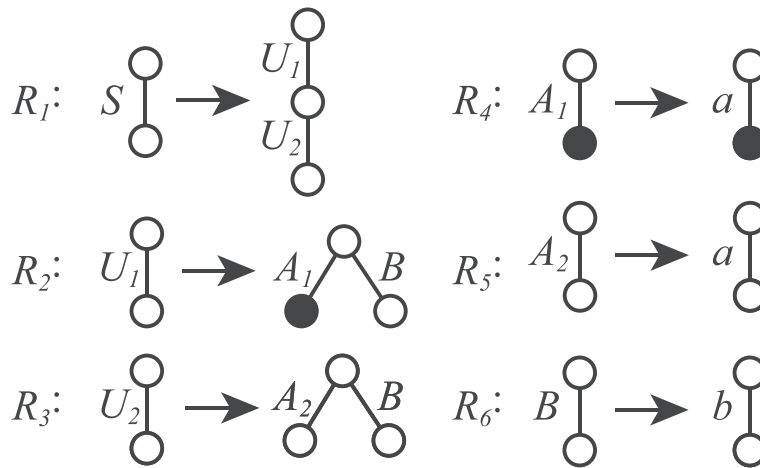
nonterminal symbols are replaced with terminal symbols, and the strings 'ab', 'aabb', 'aaabbb', etc. are generated, by this grammar. The size of the grammar is 5. In a case where  $T = \{a, b\}$ ,  $V = \{S, X, Y\}$ , and  $P = \{S \rightarrow aXb, X \rightarrow aYb, Y \rightarrow ab\}$ , only 'aaabbb' is generated from  $S$ . The size of this grammar is 8.  $X$  and  $Y$  represent 'aabb' and 'ab', respectively. The rest of this paper deals with grammars that generate a constant number of trees. Each nonterminal symbol represents a specified tree.

**Simple elementary ordered/unordered tree grammar (SEOTG/SEUTG)**

Simple elementary ordered tree grammar (SEOTG) is defined for the rooted ordered trees  $T(V, E)$ , where  $V$  is a set of vertices, and  $E$  is a set of labeled edges. If  $T$  represents a glycan, a vertex corresponds to a monosaccharide, an edge corresponds to a bond between the monosaccharides, and the enzyme involved in the biosynthesis can be represented by the label of the edge, as shown in Figure 1. As well as CFGs, a grammar of SEOTG consists of 4-tuple  $(\Sigma, \Gamma, S, \Delta)$ , where  $\Sigma$  is a set of terminal symbols,  $\Gamma$  is a set of nonterminal symbols,  $S$  is a start symbol in  $\Gamma$ , and  $\Delta$  is a set of production rules, that are classified into Horizontal Bisection (RHB), Vertical Bisection (RVB), and Name Change (RNC), as in Figure 2. It should be noted that production rules of SEOTG and SEUTG are different from construction rules of biomolecular trees. (RHB) includes three rules that an edge of nonterminal symbol  $A$  is replaced with a tree whose root is both roots of nonterminal symbols  $B$  and  $C$ .  $A$  is bisected at the root into  $B$  and  $C$ . We introduce a tag to represent the vertex connected with another tree. The first rule in (RHB) does not contain a tag, and the other rules contain a tag, respectively. (RVB) includes two rules that an edge of nonterminal symbol  $A$  is



**Figure 2** Production rules for Simple Elementary Ordered Tree Grammar (SEOTG). (RHB) Horizontal bisection rules, (RVB) Vertical bisection rules, (RNC) Replacement rules with a terminal symbol. The uppercase letters, 'A', 'B', and 'C' indicate non-terminal symbols, while the lowercase letter, 'a', indicates a terminal symbol. The circle filled in black indicates a tag.



**Figure 3** Example of SEOTG with  $(\Sigma, \Gamma, S, \Delta)$ .  $\Sigma$  indicates a set of terminal symbols,  $\Gamma$  indicates a set of nonterminal symbols  $S, U_1, U_2, A_1, A_2, B$ , and  $\Delta$  indicates the set of six production rules,  $R_1, \dots, R_6$ .

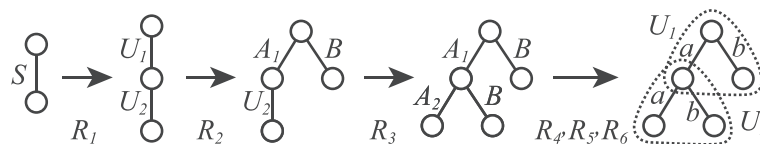
replaced with a tree in which the root is the root of nonterminal symbol  $B$ , and the root of nonterminal symbol  $C$  is attached to the tag of  $B$ .  $A$  is bisected at an internal vertex of  $A$  into  $B$  and  $C$ . (RNC) includes two rules that an edge of nonterminal symbol  $A$  is replaced with an edge of terminal symbol  $a$ . In addition, any nonterminal symbol does not appear in expansion of the symbol itself. Then, each nonterminal symbol corresponds to a subtree of a given tree  $T$ . Figure 3 shows an example of SEOTG with  $(\Sigma, \Gamma, S, \Delta)$ .  $\Delta$  is the set of six production rules,  $R_1, \dots, R_6$ , where  $R_1$  is a vertical bisection rule,  $R_2$  and  $R_3$  are horizontal bisection rules, and  $R_4, R_5, R_6$  are name change rules. Figure 4 illustrates the derivation of a tree from the start symbol  $S$  by the SEOTG. The first replacement is done by  $R_1$ , and  $U_1$  is replaced with the right-hand side of  $R_2$ . Then, the lower endpoint of  $U_1$  connects with the root of  $U_2$ , and one of leaves of the replaced tree of  $U_1$  connects with the root of  $U_2$ . A tag indicates the vertex connected with another vertex. Hence, the lower endpoint labeled with a tag in  $A_1$  connects with the root of  $U_2$ . By applying  $R_3, \dots, R_6$ , the right-most tree is generated. The trees surrounded by dotted curves are derived from nonterminal symbols  $U_1$  and  $U_2$ , respectively. It is considered in the example of Figure 1 that a terminal symbol corresponds to a biosynthesis, and a nonterminal symbol corresponds to a sequence of biosyntheses.

Simple elementary unordered tree grammar (SEUTG) is defined for rooted unordered trees in a manner similar to the SEOTG. In SEUTG, the second and third production rules with a tag in the (RVB), described in Figure 2, are equivalent to each other, as the trees are unordered.

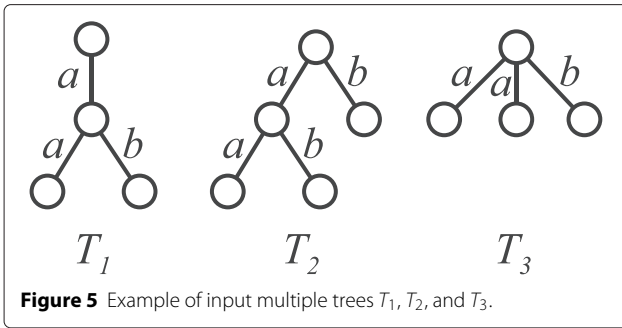
**Extension to multiple trees**

We extend the SEOTG and SEUTG to multiple trees.  $N$  is the number of given trees, and  $T_\alpha$  indicates the  $\alpha$ -th edge labeled rooted tree. The start symbol  $S$  is replaced with the set of the nonterminal symbols  $S_\alpha$ . Each tree  $T_\alpha$  is generated from  $S_\alpha$  using one grammar. Figure 5 shows an example of the input multiple trees  $T_1, T_2$ , and  $T_3$ . One of the minimum grammars generating these trees is shown in Figure 6. The size of the grammar is the total number of symbols present on the right-hand side in the production rules, i.e., 11. We minimize the number of distinct nonterminal symbols instead of the size of the grammar, as there exists the same number of production rules as the number of distinct nonterminal symbols. Figure 7 illustrates the derivation of  $T_2$  from the start symbol  $S_2$  using the grammar.  $T_2$  is the same as  $T$  in Figure 4, and we observe the modification of the derivation process by providing other similar trees.

The Euler string  $es(T)$  is used to determine if the labeled rooted trees  $T_1$  and  $T_2$  are isomorphic to each other,



**Figure 4** Derivation of tree  $T$  using the grammar, with the set of production rules shown in Figure 3. The trees surrounded by the dotted curves are derived from nonterminal symbols  $U_1$  and  $U_2$ , respectively.



**Figure 5** Example of input multiple trees  $T_1$ ,  $T_2$ , and  $T_3$ .

where  $es(T)$  for a tree  $T$  is defined by the sequence of edge labels  $l$  and its opposite  $\bar{l}$ , along the depth-first traversal of  $T$  [17]. For example, for  $T_3$  in Figure 5,  $es(T_3)$  is determined to be  $a\bar{a}a\bar{a}b\bar{b}$ . For a tagged tree, the tagged edge, labeled  $a$ , is transformed into  $a\tau\bar{a}$ , using a special symbol,  $\tau$ , which represents the tag. It is noted that for two edge labeled rooted trees  $T_1$  and  $T_2$ ,  $T_1$  is isomorphic to  $T_2$  if (and only if)  $es(T_1) = es(T_2)$ .

$U$  is assigned as the set of all Euler strings for all connected subtrees of  $N$  trees. By improving the previous formulation, we propose the following IP formulation, MinSEOTGMul, for the direct determination of the minimum SEOTG that constructs  $N$  ordered trees  $T_\alpha$ .

$$\text{Minimize } \sum_{u \in U} p_u$$

Subject to

$$x_{\alpha,i,\epsilon,jj} = 1 \quad \text{for all } \alpha, i, j \in ch(\alpha, i) (|ch(\alpha, j)| = 0), \quad (1)$$

$$x_{\alpha,i,jjj} = 1 \quad \text{for all } \alpha, i, j \in ch(\alpha, i) (|ch(\alpha, j)| > 0), \quad (2)$$

$$x_{\alpha,1,\epsilon,lch(\alpha,1),rch(\alpha,1)} = 1 \quad \text{for all } \alpha, \quad (3)$$

$$x_{\alpha,i,\epsilon,h,k} \leq \sum_{l=h}^{k-1} y_{\alpha,i,\epsilon,h,l,k} + \sum_{t \in I(T_{\alpha,i,\epsilon,h,k})} z_{\alpha,i,\epsilon,h,k,t}$$

for all  $\alpha, i, h \leq k \in ch(\alpha, i)$ , (4)

$$y_{\alpha,i,\epsilon,h,l,k} \leq \frac{1}{2} (x_{\alpha,i,\epsilon,h,l} + x_{\alpha,i,\epsilon,l+1,k})$$

for all  $\alpha, i, h \leq l < k \in ch(\alpha, i)$ , (5)

$$z_{\alpha,i,\epsilon,h,k,t} \leq \frac{1}{2} (x_{\alpha,i,t,h,k} + x_{\alpha,t,\epsilon,lch(\alpha,t),rch(\alpha,t)})$$

for all  $\alpha, i, h \leq k \in ch(\alpha, i), t \in I(T_{\alpha,i,\epsilon,h,k})$ , (6)

$$x_{\alpha,i,j,h,k} \leq \sum_{l=h}^{k-1} y_{\alpha,i,j,h,l,k} + \sum_{t \in anc(\alpha,j)} z_{\alpha,i,j,h,k,t}$$

for all  $\alpha, i, h \leq k \in ch(\alpha, i), j \in I(T_{\alpha,i,\epsilon,h,k})$ , (7)

$$y_{\alpha,i,j,h,l,k} \leq \frac{1}{2} (x_{\alpha,i,\epsilon,h,l} + x_{\alpha,i,j,l+1,k})$$

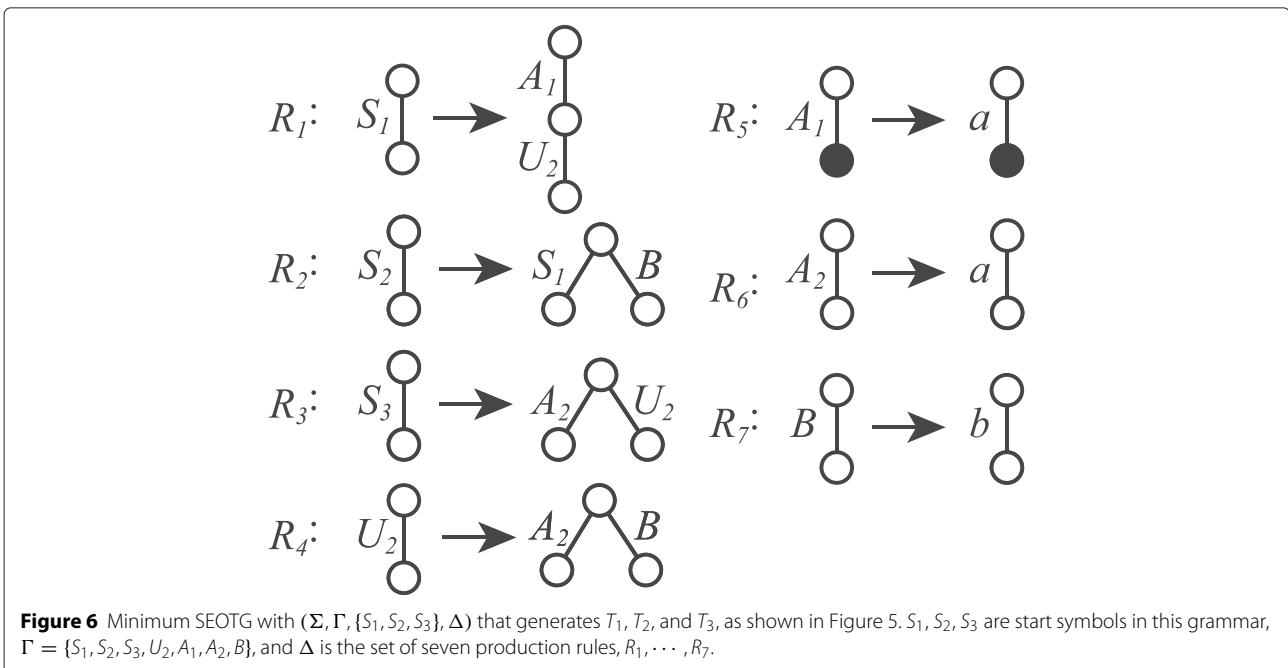
for all  $\alpha, i, h \leq l < k \in ch(\alpha, i)$ , (8)  
 $j \in I(T_{\alpha,i,\epsilon,l+1,k})$ ,

$$y_{\alpha,i,j,h,l,k} \leq \frac{1}{2} (x_{\alpha,i,j,h,l} + x_{\alpha,i,\epsilon,l+1,k})$$

for all  $\alpha, i, h \leq l < k \in ch(\alpha, i)$ , (9)  
 $j \in I(T_{\alpha,i,\epsilon,h,l})$ ,

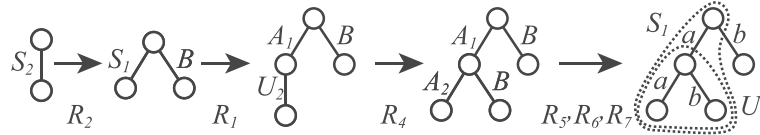
$$z_{\alpha,i,j,h,k,t} \leq \frac{1}{2} (x_{\alpha,i,t,h,k} + x_{\alpha,t,j,lch(\alpha,t),rch(\alpha,t)})$$

for all  $\alpha, i, h \leq k \in ch(\alpha, i), j \in I(T_{\alpha,i,\epsilon,h,k})$ , (10)  
 $t \in anc(\alpha, j)$ ,



**Figure 6** Minimum SEOTG with  $(\Sigma, \Gamma, \{S_1, S_2, S_3\}, \Delta)$  that generates  $T_1, T_2$ , and  $T_3$ , as shown in Figure 5.  $S_1, S_2, S_3$  are start symbols in this grammar,  $\Gamma = \{S_1, S_2, S_3, U_2, A_1, A_2, B\}$ , and  $\Delta$  is the set of seven production rules,  $R_1, \dots, R_7$ .





**Figure 7** Derivation of the tree  $T_2$  shown in Figure 5 by the grammar, using the set of production rules shown in Figure 6. The trees surrounded by dotted curves are derived from nonterminal symbols  $S_1$  and  $U_2$ , respectively.

$$s_u \leq p_u < 1 + s_u \quad \text{for all } u \in U, \tag{11}$$

$$s_u = \frac{1}{|\{(\alpha, i, j, h, k) | es(T_{\alpha, i, j, h, k}) = u\}| \sum_{\{(\alpha, i, j, h, k) | es(T_{\alpha, i, j, h, k}) = u\}} x_{\alpha, i, j, h, k}} \tag{12}$$

Here,  $lch(\alpha, i)$ ,  $rch(\alpha, i)$ , and  $ch(\alpha, i)$  denote the leftmost child of the vertex  $v_i$  in  $T_\alpha$ , the rightmost child of  $v_i$  in  $T_\alpha$ , and the set of child vertices of  $v_i$  in  $T_\alpha$ , respectively.  $T_{\alpha, i, t, h, k}$  denotes the subtree rooted at vertex  $v_i$ , with the child vertices  $v_j$  ( $h \leq j \leq k$ ) and  $v_t$  labeled with a tag in  $T_\alpha$ , which does not have a tag when  $t = \epsilon$  (Figure 8).  $I(T)$  denotes the set of internal vertices, except for the root and leaves of tree  $T$ .  $anc(\alpha, j)$  denotes the set of ancestor vertices of  $v_j$ , where  $j \notin anc(j)$  and  $anc(\epsilon) = \emptyset$ .

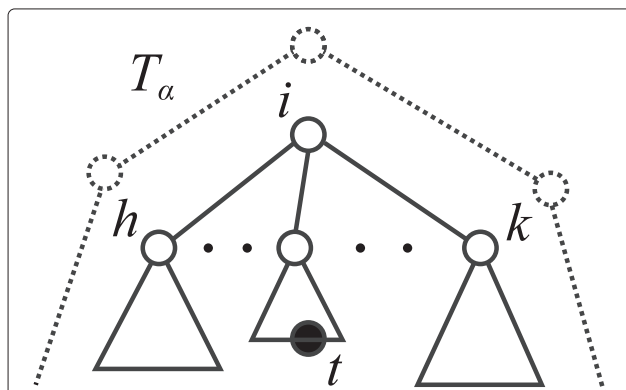
In MinSEOTGMul, the variable  $p_u$  is equated to 1 if a nonterminal symbol that corresponds to the subtree represented by Euler string  $u$  appears in the grammar; otherwise, the variable is equated to 0. MinSEOTGMul minimizes the sum of  $p_u$ , i.e., the number of distinct nonterminal symbols appeared in the output grammar as a result. The variable  $x_{\alpha, i, t, h, k}$  takes on a value of 1, if the subtree  $T_{\alpha, i, t, h, k}$  is constructed by the grammar; otherwise, the value of this variable remains 0. In Eqs. (1) and (2),  $x_{\alpha, i, \epsilon, j}$  and  $x_{\alpha, i, j, j}$  correspond to an edge in  $T_\alpha$ , and each edge in the  $N$  trees is always constructed according to a production rule of (RNC).  $x_{\alpha, 1, \epsilon, lch(\alpha, 1), rch(\alpha, 1)}$  corresponds to the entire  $\alpha$ -th tree  $T_\alpha$ , where the root of each

tree is numbered as 1. Eq. (3) represents that MinSEOTGMul requires that all  $N$  trees  $T_\alpha$  are constructed using the grammar.

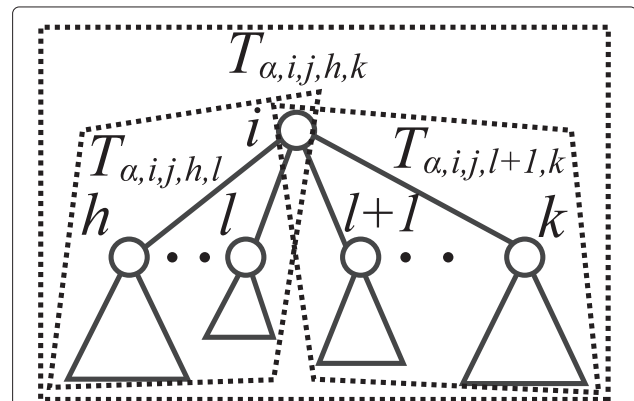
The variable  $y_{\alpha, i, j, h, l, k}$  takes on a value of 1 if  $T_{\alpha, i, j, h, k}$  is constructed from  $T_{\alpha, i, j, h, l}$  and  $T_{\alpha, i, j, l+1, k}$  using an (RHB) production rule; otherwise, the value is maintained at 0 (Figure 9). The variable  $z_{\alpha, i, j, h, k, t}$  is denoted as 1 if  $T_{\alpha, i, j, h, k}$  is constructed from  $T_{\alpha, i, t, h, k}$  and  $T_{\alpha, t, j, lch(\alpha, t), rch(\alpha, t)}$  using an (RVB) production rule; otherwise, the value is retained as 0 (Figure 10). Eqs. (4) and (7) indicate that the subtree  $T_{\alpha, i, j, h, k}$  is constructed by at least one established production rule of (RHB) and (RVB) in the grammar. Eqs. (5), (6), (8), (9), and (10) indicate that a production rule of (RHB) and (RVB) becomes a candidate rule in the grammar when both of the two source subtrees are constructed.

The variable  $s_u$  is defined by Eq. (12), and takes on a real value of  $0 \leq s_u \leq 1$ . If at least one subtree  $T_{\alpha, i, j, h, k}$  whose Euler string is  $u$ , i.e.,  $es(T_{\alpha, i, j, h, k}) = u$ , is constructed, then  $s_u > 0$ . Based on Eq. (11),  $p_u$  takes on a value of 1. It means that one nonterminal symbol corresponding to the subtree appears in the grammar. Conversely, when any subtree whose Euler string is  $u$  is not constructed, then  $s_u = 0$ ,  $p_u$  takes on a value of 0, and a nonterminal symbol is not generated. In our previous study, unnecessary nonterminal symbols could be generated, and made it difficult to find the minimum number of nonterminal symbols.

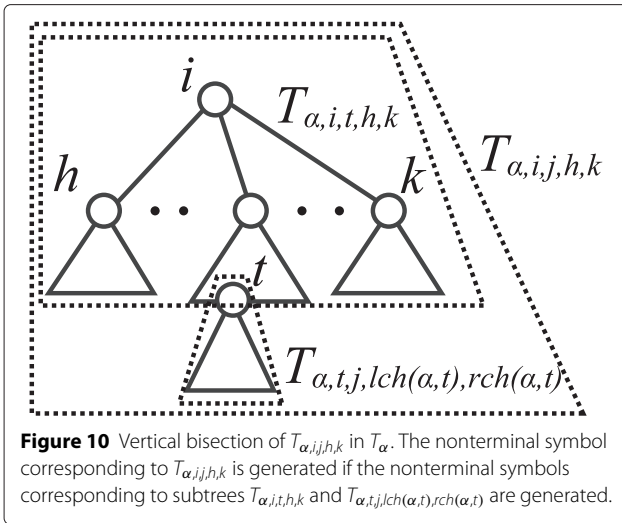
For unordered trees, we propose the following IP formulation, MinSEUTGMul, to determine the minimum



**Figure 8** Illustration of subtree  $T_{\alpha, i, t, h, k}$  in  $T_\alpha$ .  $T_{\alpha, i, t, h, k}$  denotes the subtree rooted at vertex  $v_i$  having the child vertices  $v_j$  ( $h \leq j \leq k$ ) and vertex  $v_t$  labeled with a tag in  $T_\alpha$ .



**Figure 9** Horizontal bisection of  $T_{\alpha, i, j, h, k}$  in  $T_\alpha$ . The nonterminal symbol corresponding to  $T_{\alpha, i, j, h, k}$  is generated if the nonterminal symbols corresponding to subtrees  $T_{\alpha, i, j, h, l}$  and  $T_{\alpha, i, j, l+1, k}$  are generated.



SEUTG for constructing  $N$  unordered trees  $T_{\alpha}$ , in a manner similar to that used for ordered trees.

$$\text{Minimize } \sum_{u \in U} p_u$$

Subject to

$$x_{\alpha,i,\epsilon,\{j\}} = 1 \quad \text{for all } \alpha, i, j \in ch(\alpha, i) (|ch(\alpha, j)| = 0),$$

$$x_{\alpha,i,j,\{j\}} = 1 \quad \text{for all } \alpha, i, j \in ch(\alpha, i) (|ch(\alpha, j)| > 0),$$

$$x_{\alpha,1,\epsilon,ch(\alpha,1)} = 1 \quad \text{for all } \alpha,$$

$$x_{\alpha,i,\epsilon,C} \leq \sum_{C'(\neq \emptyset) \subset C} y_{\alpha,i,\epsilon,C',C-C'} + \sum_{t \in I(T_{\alpha,i,\epsilon,C})} z_{\alpha,i,\epsilon,C,t}$$

$$\text{for all } \alpha, i, C \subseteq ch(\alpha, i),$$

$$y_{\alpha,i,\epsilon,C',C-C'} \leq \frac{1}{2}(x_{\alpha,i,\epsilon,C'} + x_{\alpha,i,\epsilon,C-C'})$$

$$\text{for all } \alpha, i, C \subseteq ch(\alpha, i), C'(\neq \emptyset) \subset C,$$

$$z_{\alpha,i,\epsilon,C,t} \leq \frac{1}{2}(x_{\alpha,i,t,C} + x_{\alpha,t,\epsilon,ch(\alpha,t)})$$

$$\text{for all } \alpha, i, C \subseteq ch(\alpha, i), t \in I(T_{\alpha,i,\epsilon,C}),$$

$$x_{\alpha,i,j,C} \leq \sum_{C'(\neq \emptyset) \subset C} y_{\alpha,i,j,C',C-C'} + \sum_{t \in anc(\alpha,j)} z_{\alpha,i,j,C,t}$$

$$\text{for all } \alpha, i, C \subseteq ch(\alpha, i), j \in I(T_{\alpha,i,\epsilon,C}),$$

$$y_{\alpha,i,j,C',C-C'} \leq \frac{1}{2}(x_{\alpha,i,\epsilon,C'} + x_{\alpha,i,j,C-C'})$$

$$\text{for all } \alpha, i, C \subseteq ch(\alpha, i), j \in I(T_{\alpha,i,\epsilon,C}), C'(\neq \emptyset) \subset C,$$

$$z_{\alpha,i,j,C,t} \leq \frac{1}{2}(x_{\alpha,i,t,C} + x_{\alpha,t,j,ch(\alpha,t)})$$

$$\text{for all } \alpha, i, C \subseteq ch(\alpha, i), j \in I(T_{\alpha,i,\epsilon,C}), t \in anc(\alpha, j),$$

$$s_u \leq p_u < 1 + s_u \quad \text{for all } u \in U,$$

$$s_u = \frac{1}{|\{(\alpha, i, j, C) | es(T_{\alpha,i,j,C}) = u\}|} \sum_{\{(\alpha, i, j, C) | es(T_{\alpha,i,j,C}) = u\}} x_{\alpha,i,j,C}.$$

The horizontal bisection rules of (RHB) split the set  $C$  of child vertices of the vertex  $v_i$  into a subset  $C'$ , and the remaining vertices  $C - C'$ .  $T_{\alpha,i,j,C}$  indicates that the subtree rooted at vertex  $v_i$  has a set  $C$  of the child vertices. The variables  $x_{\alpha,i,j,C}$ ,  $y_{\alpha,i,j,C',C-C'}$ , and  $z_{\alpha,i,j,C,t}$  are used in a manner similar to  $x_{\alpha,i,j,h,k}$ ,  $y_{\alpha,i,j,h,k}$ , and  $z_{\alpha,i,j,h,k,t}$  in MinSEOTGMul, respectively.

It should be noted that the IP formulations MinSEOTGMul and MinSEUTGMul can output multiple grammars with the minimum number of nonterminal symbols. Figure 11 displays such an example, where the grammars  $G_1$  and  $G_2$  generate the tree  $T$ , and the number of nonterminal symbols of  $G_1$  ( $G_2$ ) is 3. The first production rule  $R_1$  of  $G_1$  is different from the  $R_1$  of  $G_2$ . By providing more such trees, the number of the minimum grammars can be reduced to almost one.

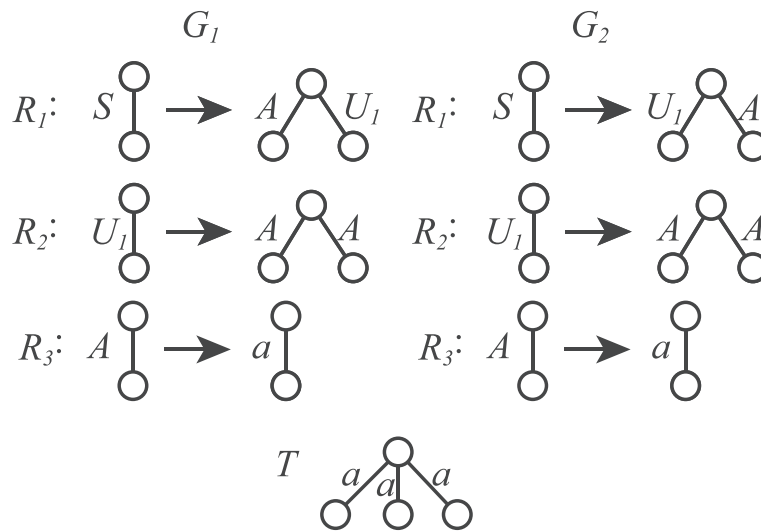
## Results and discussion

### Tree representation of glycans and RNAs

The proposed methods MinSEOTGMul and MinSEUTGMul were evaluated by preparing two types of biological data, glycans and RNA secondary structures, which were dealt with as unordered and ordered trees, respectively. For this analysis, we utilized 16 glycans, G02677, G03655, G03661, G03664, G03678, G03687, G04186, G04458, G04695, G04802, G05058, G05226, G05256, G05988, G07243, and G09054, from the KEGG Glycan database [20]. As glycans are regarded as vertex labeled rooted trees, wherein each vertex is a monosaccharide, the glycans were transformed into edge labeled rooted trees, wherein each edge is labeled with a label of its lower vertex.

In addition, 24 RNA secondary structures belonging to distinct RNA families were taken from the Rfam database [21], as shown in Additional file 1: Table S1 on our supplementary web site; these were transformed into rooted ordered trees. For this, one sequence was selected from multiple sequence alignments of each RNA family, as our method requires edge labels, i.e., bases. RNA secondary structures consist of base pairs with hydrogen bonds, and group binding, such as bulges and hairpin loops (as seen in Figure 12 (a)). There are several representations of trees for RNA secondary structures. An RNA secondary structure can be represented as an ordered rooted tree, by labeling the vertices with unpaired loops and the edges with paired bases [22]; this structure can be represented as an ordered rooted tree by labeling the vertices with hairpin loops, internal loops, bulges, and paired bases [7,23]. Chen and Zhang represented an RNA secondary structure using a paired base and a leaf, corresponding to an internal vertex and an unpaired base, respectively [8]. In our implementation, the representation by [8] was modified by eliminated vertices other than those corresponding to paired bases; in





**Figure 11** Minimum SEOTGs  $G_1$  and  $G_2$  that generate tree  $T$ . The number of nonterminal symbols of  $G_1$  ( $G_2$ ) is 3. The first production rule  $R_1$  of  $G_1$  is different from  $R_1$  of  $G_2$ .

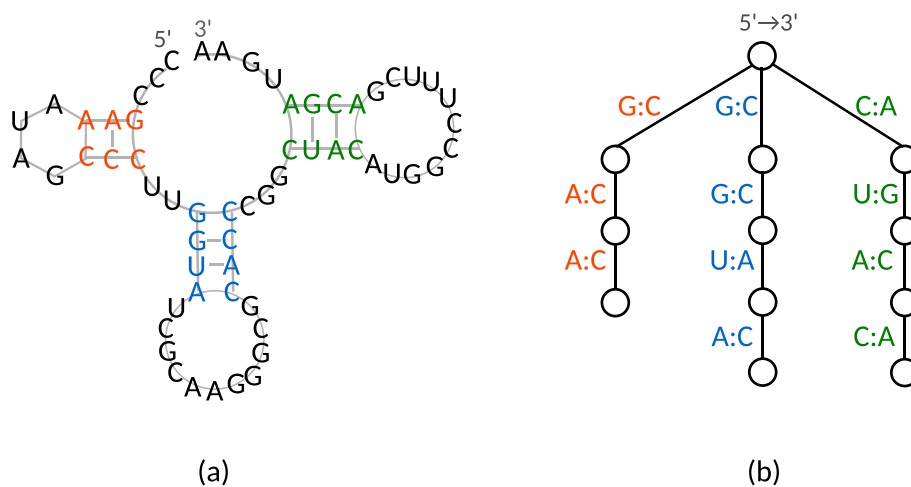
addition, the vertex labeled tree was transformed into an edge labeled tree in a manner similar to the glycans. Figure 12 illustrates the transformation, wherein a paired base was transformed into an edge, labeled with its base pair. It is noted that the edges in this representation are ordered by following the 5'-3' direction of the RNA sequence.

The CPLEX Optimization Studio (version 12.5) was used to solve integer programming using a linux operating system. The source code to transform given multiple trees into the proposed IP formulations is available at our

supplementary web site on <http://sunflower.kuicr.kyoto-u.ac.jp/tyoyo/treecomp/>.

### Minimum grammars for glycans and RNAs

We applied the proposed method MinSEUTGMul for the glycans and their several combinations. Table 1 shows the minimum number of nonterminal symbols of SEUTG for glycan unordered trees. In all cases, the minimum number of nonterminal symbols for multiple trees of glycans was lower than the sum of the minimum numbers for its single trees. For example, the minimum number of



**Figure 12** Illustration of RNA tree representation. **(a)** An artificial RNA secondary structure. **(b)** The tree representation of (a) in which paired bases are extracted by following the sequence order from 5' to 3', where bases in loops are removed. Base pairs belonging to the same secondary structure are filled in the same color.

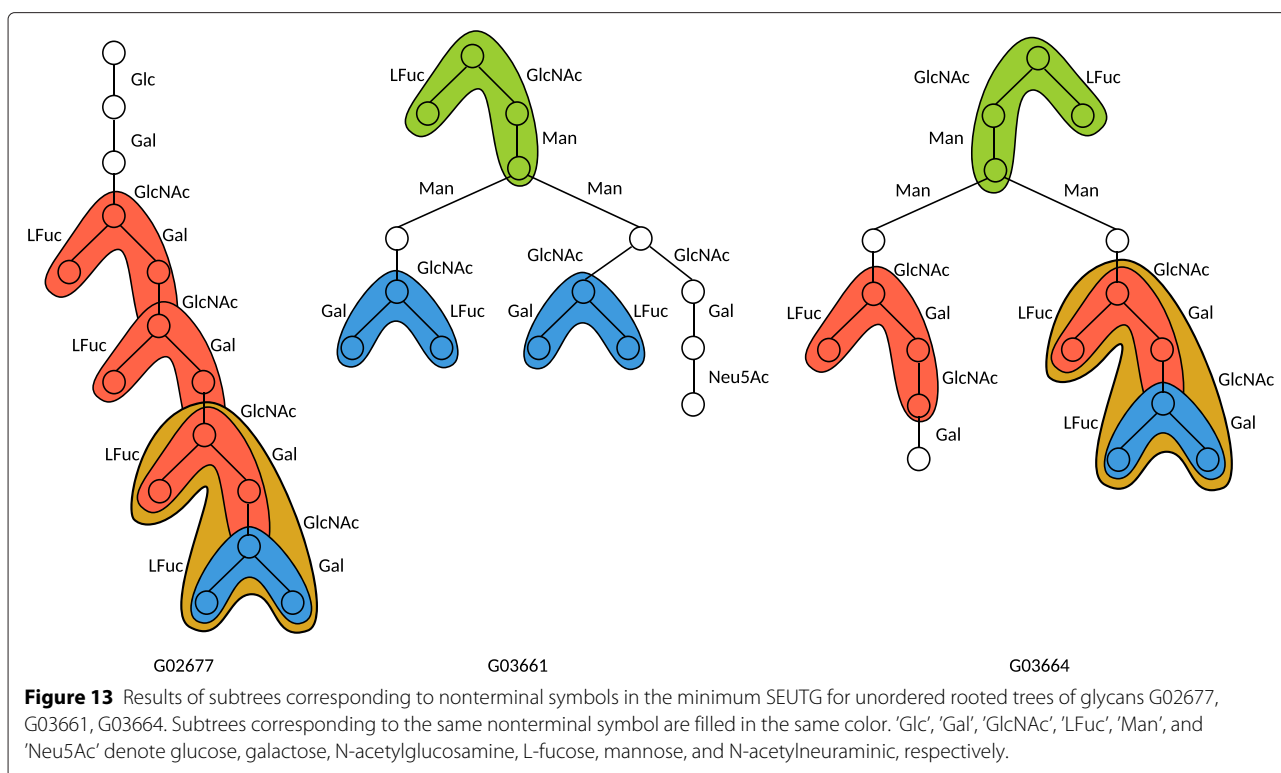
**Table 1 Results on the minimum number of nonterminal symbols by MinSEUTGMul for glycan unordered trees**

Glycan	#vertices	Sum	Min	Time (sec)	Memory (MB)
G02677	15		13	4.2	173.75
G03655	34		26	35.94	755.59
G03661	15		17	1.76	171.56
G03664	17		16	4.17	178.46
G03678	18		20	2.33	212.38
G03687	28		17	77.97	930.05
G04186	20		19	2.64	216.28
G04458	21		9	3.96	207.81
G04695	20		17	2.31	196.89
G04802	20		15	5.39	189.74
G05058	25		16	5.65	221.36
G05226	19		14	5.24	173.63
G05256	25		11	4.62	222.64
G05988	19		18	5.27	204.9
G07243	18		15	4.58	171.89
G09054	31		17	5.06	290.62
G02677, G03661, G03664	47	46	31	268.06	948.56
G03661, G03678, G04186	53	56	44	26.97	681.83
G03678, G03687, G04186	66	56	52	66.45	791.04
G04695, G05058, G05226	64	47	44	137.87	1080.9
G04458, G04695, G04802	61	41	28	408.87	3088.68
G05988, G07243, G09054	68	50	43	68.29	942.43
G02677, G03661, G03664, G03678	65	66	47	96.14	923.62
G02677, G03661, G04458, G07243	69	54	39	8748	16814.68
G03661, G03678, G04186, G05226	72	70	49	56.62	1068.76
G04458, G04695, G04802, G05226	80	55	40	97.1	954.66
G02677, G03661, G03664, G03678, G07243	83	81	52	181.19	955.31
G03661, G03678, G04186, G05226, G05988	91	88	51	51.52	804.57
G03664, G04802, G05256, G05988, G07243	99	75	49	425.89	1717.89
G04458, G04695, G04802, G05226, G05988	99	73	41	164.41	1119.74
G02677, G03661, G03664, G03678, G05226, G07243	102	95	53	220.73	976.34
G03661, G03678, G04186, G05226, G05988, G07243	109	103	53	50.42	1018.3

'#vertices' denotes the total number of vertices in glycan trees. 'sum' denotes the sum of the minimum numbers for single glycans in a combination, and is omitted for single glycans. 'min' denotes the minimum number for all glycans in a combination. 'time' denotes the execution time in seconds. 'memory' denotes the memory usage in mega bytes.

nonterminal symbols for G02677, G03661, and G03664 was 31, which was lower than the sum of the minimum numbers  $13 + 17 + 16 = 46$ . This suggests that our method successfully determined several common rules among the combination of glycans, and that the compression of several glycans together is better than that of each individual glycan. The execution time and the memory usage by the IP solver, for multiple trees with over 100 vertices in our experiments, were observed to be less

than 10 minutes and 4G bytes, respectively (except in the case of G02677, G03661, G04458, and G07243). These compression sizes can be used to estimate the similarities between glycan structures. If the compression size of two glycans is smaller than the sum of compression sizes of the individual glycans, these glycan structures are considered to be similar. Figure 13 shows the subtrees corresponding to the nonterminal symbols contained within the minimum SEUTG for the unordered rooted trees of



the glycans G02677, G03661, and G03664; the subtrees corresponding to the same nonterminal symbol are filled with the same color, and a portion of the nonterminal symbol is shown. The nonterminal symbol colored blue appeared in all three glycans, while those colored green and red appeared in two glycans. The nonterminal symbol colored brown consisted of the nonterminal symbols colored red and blue. This implies that the hierarchical structures contained within the glycans beyond the frequent patterns can be extracted using the developed methods.

The proposed method MinSEOTGMul was applied to the RNA secondary structures and their several combinations. Table 2 shows the minimum number of nonterminal symbols of SEOTG for RNA ordered trees. In all cases, the minimum number of nonterminal symbols for multiple trees of RNA was lower than the sum of minimum numbers for its single trees, similar to the glycans. The execution time and the memory usage by the IP solver for multiple trees in our experiments were between two seconds and six hours, and between 260 Mbytes and 37 Gbytes, respectively. Figure 14 shows the subtrees corresponding to nonterminal symbols in the minimum SEOTG for ordered rooted trees of the RNA secondary structures RF00002 and RF00008. Figure 15 shows the original RNA secondary structures of RF00002 and RF00008, and the base pairs corresponding to nonterminal symbols in

the minimum SEOTG. We also observed the hierarchical structure of the nonterminal symbols, colored blue and brown.

We examined the alternative approach that transforms multiple trees into a single tree and applies our previous methods. For the set of glycans G02677, G03661, and G03664, the alternative method output the existence of the SEUTG grammar with size 39 in 8.83 seconds. However, we could not obtain the result for size 38 within 24 hours, and could not determine the minimum number of nonterminal symbols. For the set of RNAs RF00002, and RF00004, the method output the existence of the SEOTG grammar with size 42 in 5.01 seconds. However, we could not obtain the result for size 41. We can see that the proposed methods are more efficient than the previous methods.

### Conclusions

We proposed novel integer programming-based methods MinSEOTGMul and MinSEUTGMul to determine the minimum simple elementary ordered and unordered tree grammars (SEOTG and SEUTG) for multiple ordered and unordered trees, respectively. These could be directly applied to the determination of the minimum grammar, unlike our previously proposed methods. We applied MinSEUTGMul to several unordered trees transformed from glycans, and their combinations; MinSEOTGMul

**Table 2 Results on the minimum number of nonterminal symbols by MinSEOTGMul for RNA ordered trees**

RNA	#vertices	Sum	Min	Time (sec)	Memory (MB)
RF00002	20		26	57.77	593.89
RF00003	23		31	4836	8186.96
RF00004	14		18	0.5	194.7
RF00005	22		24	411.9	957.51
RF00007	16		21	4.27	261.98
RF00008	14		18	7.51	212.37
RF00016	8		11	0.13	22.49
RF00029	20		26	1010.53	1592.98
RF00032	7		9	0.06	22.18
RF00050	24		25	1993.69	1721.94
RF00072	15		19	8.04	579.91
RF00101	18		22	44.38	562.54
RF00137	8		11	0.09	22.56
RF00166	11		13	0.09	23.3
RF00167	21		24	2099.08	2025.98
RF00234	21		24	531.7	920.3
RF00360	7		9	0.05	22.24
RF00442	18		21	23.99	585.23
RF00517	13		14	0.58	170.37
RF00519	13		10	0.28	25.9
RF01054	18		19	9.47	502.48
RF01829	9		11	0.08	22.47
RF01850	20		21	68.63	710.26
RF01851	21		23	460.43	1066.23
RF00002, RF00004	34	44	37	4329	3962.39
RF00002, RF00008	34	44	36	1861.88	2702.14
RF00003, RF00004	37	49	43	5600	9553.93
RF00004, RF00007	30	39	34	2104.57	3023.84
RF00004, RF00016	22	29	23	2.04	264.36
RF00005, RF00016	30	35	30	61.19	5326.1
RF00007, RF00008	30	39	34	922.44	4152.85
RF00007, RF00032	23	30	26	2.46	282.71
RF00016, RF00029	28	37	31	106.51	13453.93
RF00016, RF00072	23	30	21	5.76	486.06
RF00029, RF00360	27	35	31	142.6	3537.38
RF00032, RF00050	31	34	30	3405.65	5729.7
RF00050, RF00137	32	36	35	214.27	3325.72
RF00050, RF01829	33	36	29	7958	11276
RF00072, RF00137	23	30	28	6.2	527.51
RF00072, RF01829	24	30	27	6.49	542.42
RF00101, RF01829	27	33	29	244.58	1004.71
RF00166, RF00167	32	37	32	19138	36018.49
RF00166, RF01054	29	32	28	151.13	851.54

**Table 2 Results on the minimum number of nonterminal symbols by MinSEOTGMul for RNA ordered trees**

*(Continued)*

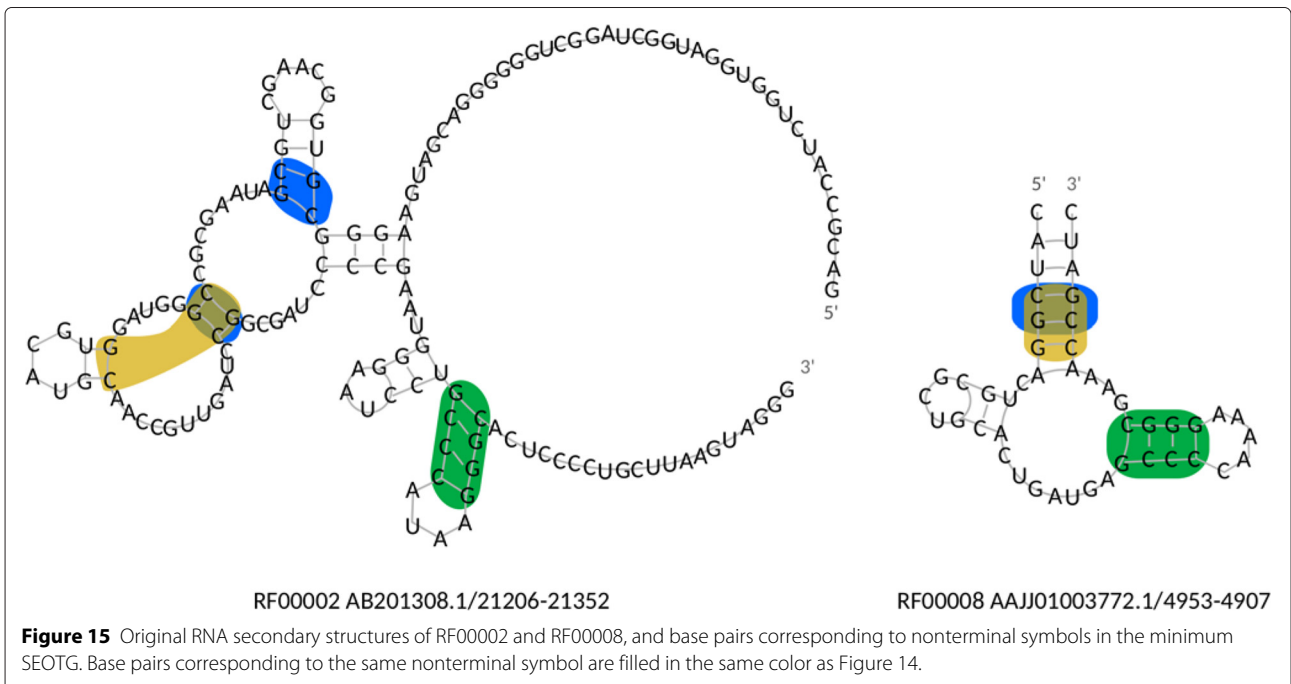
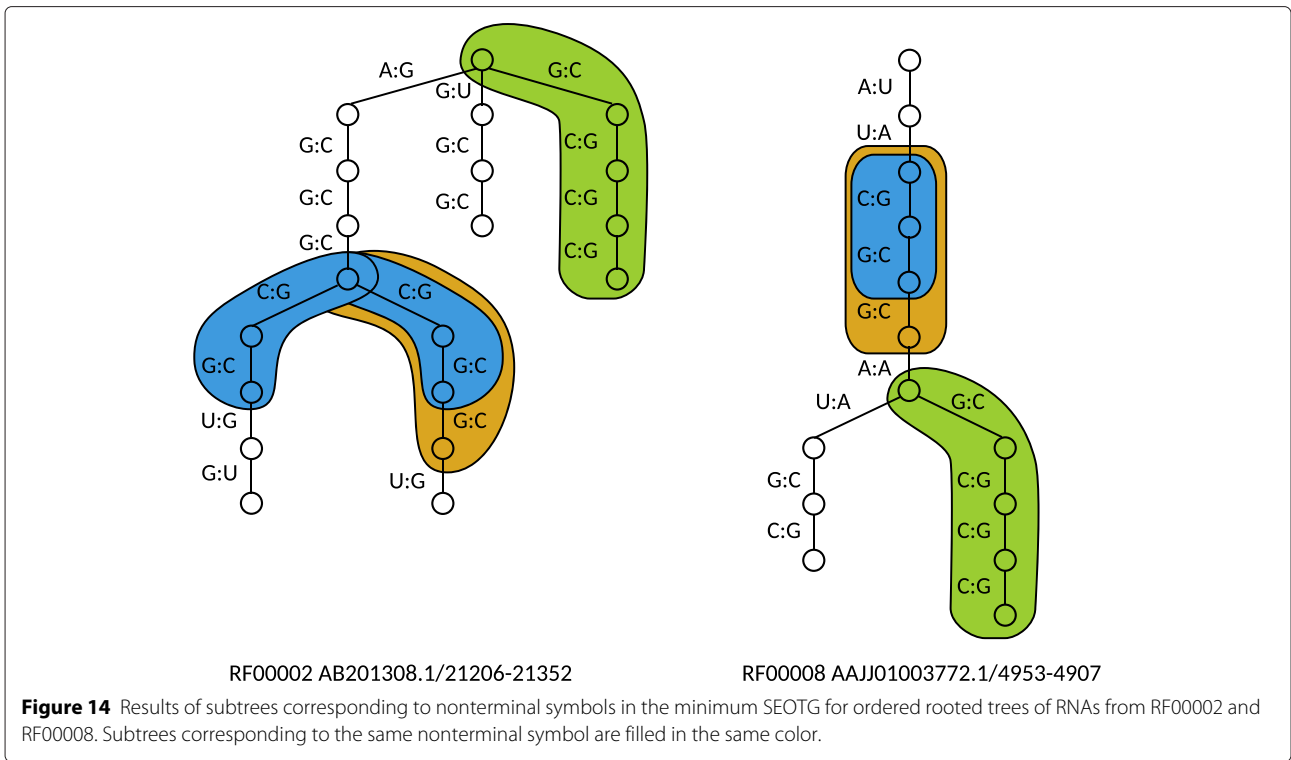
RF00360, RF00442	25	30	27	47.75	593.08
RF00442, RF00519	31	31	26	81.07	759.43
RF00517, RF00519	26	24	22	3.38	403.29
RF01850, RF01851	41	44	23	220.6	1290.55
RF00002, RF00008, RF00032	41	53	40	2209.81	4734.47
RF00002, RF00004, RF00016	42	55	41	10189	17355.98
RF00004, RF00007, RF00016	38	50	38	4605	8973.44
RF00004, RF00008, RF00032	35	45	32	135.14	733.37
RF00007, RF00008, RF00032	37	48	38	821.59	2310.18
RF00007, RF00016, RF00032	31	41	33	66.79	646.84
RF00016, RF00072, RF00137	31	41	30	4.56	353.48
RF00032, RF00137, RF00234	36	44	39	601.1	2105.12
RF00137, RF01850, RF01851	49	55	32	239.55	1472.98
RF00517, RF00519, RF01054	44	43	35	3020.41	6465.58
RF00004, RF00016, RF00072, RF00137	45	59	41	976.24	2862.5

#vertices' denotes the total number of vertices in trees transformed from RNAs. 'sum' denotes the sum of the minimum numbers for single RNAs in a combination, and is omitted for single RNAs. 'min' denotes the minimum number for all RNAs in a combination. 'time' denotes the execution time in seconds. 'memory' denotes the memory usage in mega bytes.

was applied to several ordered trees transformed from RNA secondary structures, and their combinations. In all cases, the minimum number of nonterminal symbols in the grammars used in the construction of multiple trees was lower than the sum of minimum numbers in the grammars used to construct the single trees. This suggests that the proposed methods were successful in determining several common rules for glycans and RNA. In addition, several results of the minimum grammars for multiple trees of glycans and RNA reveal that our methods can provide clues towards extracting the hierarchical structures contained within tree-structured biological data, beyond the frequent patterns.

In our experiments, the execution time and the memory usage for a set of trees required six hours and 37GBytes, respectively. To obtain the minimum SEOTG and SEUTG for more trees including more complicated trees, we need to further improve the efficiency.

In this study, we utilized the minimum grammar for extraction of common construction rules among multiple distinct trees. However, the proposed methods can be used for data compression. Furthermore, the execution times of some operations can be decreased to multiple trees by applying the operations to the previously obtained minimum grammar. In the future, we would like to apply our methods to more glycans, RNA, and other tree-structured biological data.



## Additional file

**Additional file 1: Table S1.** RNA sequences used in our experiments.

### Competing interests

The authors declare that they have no competing interests.

### Authors' contributions

YZ and MH developed and implemented the proposed methods, and drafted the manuscript. YC, JH, and TA participated in the discussions during the development of the methods and helped draft the manuscript. All authors have read and approved the final manuscript.

### Acknowledgments

This work was partially supported by Grants-in-Aid #26240034, #24500361, and #25-2920 from MEXT, Japan.

Received: 26 December 2014 Accepted: 30 March 2015

Published online: 24 April 2015

### References

- Campbell JA, Davies GJ, Bulone V, Henrissat B. A classification of nucleotide-diphospho-sugar glycosyltransferases based on amino acid sequence similarities. *Biochem J.* 1997;326:929–42.
- Hizukuri Y, Yamanishi Y, Nakamura O, Yagi F, Goto S, Kanehisa M. Extraction of leukemia specific glycan motifs in humans by computational glycomics. *Carbohydr Res.* 2005;340:2270–78.
- Aoki-Kinoshita KF, Ueda N, Mamitsuka H, Kanehisa M. ProfilePSTMM: capturing tree-structure motifs in carbohydrate sugar chains. *Bioinformatics.* 2006;22:25–34.
- Hashimoto K, Takigawa I, Shiga M, Kanehisa M, Mamitsuka H. Mining significant tree patterns in carbohydrate sugar chains. *Bioinformatics.* 2008;24:167–73.
- Li L, Ching W-K, Yamaguchi T, Aoki-Kinoshita KF. A weighted q-gram method for glycan structure classification. *BMC Bioinformatics.* 2010;11(Suppl 1):S33.
- Fukagawa D, Tamura T, Takasu A, Tomita E, Akutsu T. A clique-based method for the edit distance between unordered trees and its application to analysis of glycan structures. *BMC Bioinformatics.* 2011;12(Suppl 1):S13.
- Milo N, Zakov S, Katzenelson E, Bachmat E, Dinitz Y, Ziv-Ukelson M. Unrooted unordered homeomorphic subtree alignment of rna trees. *Algorithms Mol Biol.* 2013;8:13.
- Chen S, Zhang K. An improved algorithm for tree edit distance with applications for rna secondary structure comparison. *J Comb Optimization.* 2014;27:778–97.
- Farid A, Pabst M, Schoberer J, Altmann F, Glössl J, Strasser R. Arabidopsis thaliana alpha1,2-glycosyltransferase (ALG10) is required for efficient N-glycosylation and leaf growth. *Plant J.* 2011;68:314–25.
- Charikar M, Lehman E, Liu D, Panigrahy R, Prabhakaran M, Sahai A, et al. The smallest grammar problem. *IEEE Trans Inf Theory.* 2005;51:2554–76.
- Rytter W. Application of lempel-ziv factorization to the approximation of grammar-based compression. *Theor Comput Sci.* 2003;302:211–22.
- Sakamoto H, Maruyama S, Kida T, Shimoazono S. A space-saving approximation algorithm for grammar-based compression. *IEICE Trans Inf Syst.* 2009;92-D:158–65.
- Scicluna J, de la Higuera C. Grammatical inference of some probabilistic context-free grammars from positive data using minimum satisfiability. In: 12th International Conference on Grammatical Inference. Brookline, MA: Microtome Publishing; 2014. p. 139–52.
- Busatto G, Lohrey M, Maneth S. Efficient memory representation of xml document trees. *Inf Syst.* 2008;33:456–74.
- Murakami S, Doi K, Yamamoto A. Finding frequent patterns from compressed tree-structure data. In: Proc. 11th Int. Conf. Discovery Science. Berlin Heidelberg: Springer; 2008. p. 284–95.
- Yamagata K, Uchida T, Shoudai T, Nakamura Y. An effective grammar-based compression algorithm for tree structured data. In: Proc. 13th Int. Inductive Logic Programming. Berlin Heidelberg: Springer; 2003. p. 383–400.
- Akutsu T. A bisection algorithm for grammar-based compression of ordered trees. *Inf Process Lett.* 2010;110:815–20.
- Zhao Y, Hayashida M, Akutsu T. Integer programming-based method for grammar-based tree compression and its application to pattern extraction of glycan tree structures. *BMC Bioinformatics.* 2010; 11(Suppl 11):4.
- Hopcroft JE, Motwani R, Ullman JD. Chapter 5: Context-Free Grammars and Languages. *Introduction to Automata Theory, languages, and Computation* 2nd edn. Boston: Addison-Wesley; 2001, pp. 169–218.
- Hashimoto K, Goto S, Kawano S, Aoki-Kinoshita F, Ueda N, Hamajima M et al. Kegg as a glycome informatics resource. *Glycobiology.* 2006;16(5): 63–70.
- Burge SW, Daub J, Eberhardt R, Tate J, Barquist L, Nawrocki EP, et al. Rfam 11.0: 10 years of RNA families. 2012. doi:10.1093/nar/gks1005.
- Shapiro BA. An algorithm for comparing multiple rna secondary structures. *Comput Appl Biosci.* 1988;4:387–93.
- Höchsmann M, Voss B, Giegerich R. Pure multiple rna secondary structure alignments: a progressive profile approach. *IEEE/ACM Trans Comput Biol Bioinformatics.* 2004;1:53–62.

**Submit your next manuscript to BioMed Central and take full advantage of:**

- Convenient online submission
- Thorough peer review
- No space constraints or color figure charges
- Immediate publication on acceptance
- Inclusion in PubMed, CAS, Scopus and Google Scholar
- Research which is freely available for redistribution

Submit your manuscript at  
www.biomedcentral.com/submit

